Sensor fusion-based approach for the field robot localization on Rovitis 4.0 vineyard robot

Jurij Rakun[1*], Matteo Pantano[2], Peter Lepej[3], Miran Lakota[1]

(1. *Faculty of Agriculture and Life Sciences, University of Maribor, Pivola 10, 2311 Hoče, Slovenia*;
2. *Az. Agricola Giorgio Pantano, Via stradelle 40, Candiana, Italy*;
3. *VISTION d.o.o., Kolodvorska ulica 24, Slovenska Bistrica, Slovenia*)

Abstract: This study proposed an approach for robot localization using data from multiple low-cost sensors with two goals in mind, to produce accurate localization data and to keep the computation as simple as possible. The approach used data from wheel odometry, inertial-motion data from the Inertial Motion Unit (IMU), and a location fix from a Real-Time Kinematics Global Positioning System (RTK GPS). Each of the sensors is prone to errors in some situations, resulting in inaccurate localization. The odometry is affected by errors caused by slipping when turning the robot or putting it on slippery ground. The IMU produces drifts due to vibrations, and RTK GPS does not return to an accurate fix in (semi-) occluded areas. None of these sensors is accurate enough to produce a precise reading for a sound localization of the robot in an outdoor environment. To solve this challenge, sensor fusion was implemented on the robot to prevent possible localization errors. It worked by selecting the most accurate readings in a given moment to produce a precise pose estimation. To evaluate the approach, two different tests were performed, one with robot localization from the robot operating system (ROS) repository and the other with the presented Field Robot Localization. The first did not perform well, while the second did and was evaluated by comparing the location and orientation estimate with ground truth, captured by a hovering drone above the testing ground, which revealed an average error of 0.005 m±0.220 m in estimating the position, and 0.6°±3.5° when estimating orientation. The tests proved that the developed field robot localization is accurate and robust enough to be used on a ROVITIS 4.0 vineyard robot.
Keywords: localization, odometry, IMU, RTK GPS, vineyard, robot, sensors fusion, ROS, precision farming
DOI: 10.25165/j.ijabe.20221506.6415

1 Introduction

The use of robotic systems in agriculture is on the rise. In recent years alone, a number of new solutions have been developed. While milking[1] and inspection[2,3] robots are already commercially available, it is still not the case for some promising solutions like robots for weeding[4], fruit picking[5], or spraying[6], which are still in the prototype phase. One of the challenges is the working environment in nature with its changing conditions that affect the performance of such robotic platforms. Hence, a robust localization algorithm[7-9] is needed, which presents a fundamental part of subsequent methods.

Localization is the process performed by a robot in order to determine its position and orientation within a certain environment, enabling the robot to perform future decisions[10]. However, the localization cannot easily be solved due to sensory uncertainties that might occur and can accumulate errors over time. A solution

to this challenge lies in the sensor fusion approach[11,12] which minimizes errors and maximizes the accuracy of the localization.

Shalal et al.[13] described an approach to use localization based on camera and laser scanner data fusion to construct a local orchard map. It does so by implementing Extended Kalman filter (EKF) to develop a local orchard map of the individual trees, which also helped to improve the precision of in row-navigation. The authors report an average error of 0.103 m for position and 3.32° for orientation.

The work from Chen et al.[14] presented a sensor fusion based approach to localize a mobile platform by using readings from four diagonally placed ultrasonic sensors and cameras. The ultrasonic sensors measure the distances to the tree trunks, while the cameras help to determine the angle at which the tree has been detected. This way an average localization error of 62 mm was achieved for the selected test cases, but no orientation accuracy was reported by the authors.

Precise localization of the mobile wheeled robot is also presented by Nemec et al.[15] which was based on the sensory fusion of odometry (ODO), visual artificial landmarks, and inertial sensors. It used simple implementation and is therefore interesting for real-time processing and low-cost hardware as it is approximately four times cheaper computationally than EKF filters and promises a Root Mean Square (RMS) error below 5 mm. However, it relied on landmarks, which are usually not present in outdoor environments. In addition, the results were calculated based on simulations and real-world testing would pose additional challenges, affecting the overall results.

As presented in the previous paragraphs, robotic systems

nowadays include several sensory systems.   These usually include inertial motion units, LiDAR systems, encoders, and global positioning systems.   Upon using the information provided by these sensors, several localization approaches can be built using one of the sensors.   The simplest is odometry which summarizes the movement of the wheels of the robot.   However, on wet soil/sand-covered surfaces or when the robot is turning, it might fail and produce errors due to the wheels slipping, which leads to incongruent encoder readings.   The inertial units include gyroscopes, accelerometers, and magnetometers to measure the angles of the rotations, and accelerations to produce speeds and relative positions of the robot while moving.   These systems are not perfect and can be influenced by other metallic objects in the proximity influencing the magnetometers, or they can be affected by noise, like the one produced by vibrations of internal combustion engines.   The third is satellite navigation systems, like RTK receivers, which are becoming cheaper and more accessible with time.   Of course, they still rely on the received signals from satellites and base stations that correct and improve the accuracy. In general, their performance is good, but in some cases such as outside interferences of the signal, canyon effects, and occlusions by trees or buildings, they will produce inaccurate or false readings. The last type of sensor is the LiDAR sensor which has its limitations in range, placement on the robot, and the number of channels.   Like this, all can fail or is insufficient at a given time, so a smart switching algorithm is required to include or exclude their readings in a given time frame.   In this work, one such approach was presented and evaluated on a prototype vineyard robot.

2   Materials and methods

2.1   Rovitis and Rovitis 4.0 robots

Rovitis is a vehicle concept for the management of grapevine fields[11] which reduces the harm that frequent contact with chemicals may lead to[17].   For example, in a single yearly production season, a vine grower may come in contact with potentially harmful products at least 16 times for every hectare.   If this is done via the robot, it reduces the exposure of the vine grower to chemicals and if this is done in autonomous mode, the vine grower may do some other work while the chemicals are applied to the plants.   All this is possible with an assembly of mechanical, mechatronics, and electrical hardware components controlled by computer programs installed on an onboard computer unit.

To build a reliable field robot localization, two different robots were used.   The original Rovitis robot[16] was used when the algorithm in this study was developed, and the Rovitis 4.0[16] was used to finetune the parameters and evaluate the results.

The original Rovitis vineyard robot was based on a 414HY Dodich excavator machine (Dodich, Italy), that was modified with variable displacement closed circuit axial piston pumps.   The human-machine interfaces were removed, in order for it to be used as a field robot.   The newer Rovitis 4.0 is based on a RoboGREEN remote-controlled platform (Energreen, Italy) with a 40 Horse Power (HP) engine, where the main difference is that it uses tracks, while the Dodich platform used wheels.   Both platforms were retrofitted with IT systems and are based on a skid-steer drive principle.   For ensuring mechanical safety, a set of mechanical bumpers were installed on both platforms with sensors mounted on the proper points of the platforms.

Both platforms include mechatronics and electrical hardware for providing a way of automatic guidance to the robots.   The onboard computational unit is in charge of the overall control, with all sensors connected to efficiently control the peripherals.   To control the platform, proportional pressure control drivers were included to regulate the amount of oil going onto the oil motors, with an electrical regulator as an interface and an electric linear actuator for throttle control.

Sensors were mounted on the platform to provide environmental input data for the control algorithms.   These include the following sensors: a Micro-Electro Mechanical System-based (MEMS-based) Phidgets spatial Inertial Motion Unit (IMU)[18], a SICK LMS111[19] for Rovitis, and a Velodyne VLP16[20] for Rovitis 4.0 LiDARs, wheel encoders, and a Piksi RTK-GPS receiver[21].

The mechanical base is controlled by the developed control algorithms installed on the onboard computational unit.   The chosen operating system is Ubuntu Linux 16.04 LTS distribution with an installed meta operating system, ROS[19].

a. Original Rovitis

b. Rovitis 4.0
Figure 1   Original Rovitis and the Rovitis 4.0 robots

2.2   Localization

The process of determining the robots' pose and orientation in space is called localization.   For the 2D case, the robot's current position p can simply be represented by a vector as,

$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad (1)$$

where, x and y are coordinates; θ is the orientation of the robot based on an initial set coordinate system.   When the robot moves and reaches a new coordinate in space, going from p_n to p_{n+1}, it might have a new orientation as shown by Equation (2).

$$p_{n+1} = p_n + \begin{bmatrix} x_n \\ y_n \\ \theta_n \end{bmatrix} \qquad (2)$$

where, x_n and y_n are the current movements along X- and Y-axes, accordingly; θ_n represents the change of orientation from the last calculation of localization or step n.   The parameters used for the new step can be produced by using different sensors, as shown by Equation (3).

$$\begin{bmatrix} x_n \\ y_n \\ \theta_n \end{bmatrix} = a \cdot \begin{bmatrix} x_{ODO} \\ y_{ODO} \\ \theta_{ODO} \end{bmatrix} + b \cdot \begin{bmatrix} x_{IMU} \\ y_{IMU} \\ \theta_{IMU} \end{bmatrix} + c \cdot \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ \theta_{GPS} \end{bmatrix} \qquad (3)$$

where, weights a, b, and c can use all of the sensors equally, put an emphasis on one and (partly) discard the other(s), or simply enable the best one in a given situation, as shown in Section 3.

With each iteration of localization, an error is produced as a difference between the accurate and actual parameters caused by roundup and measurement errors (E_n) of the sensors shown in Equation (4).

$$E_n = | p_{accurate} - p_n | \qquad (4)$$

where, $p_{accurate}$ represents the true coordinates and orientation of the robot. In order to achieve the most accurate localization E_n must be minimal as shown by Equation (5).

$$E_n = \min(| p_{accurate} - p_{ODO} |, \ | p_{accurate} - p_{IMU} |, \ | p_{accurate} - p_{GPS} |) \quad (5)$$

where, p_{ODO}, p_{IMU}, and p_{GPS} correspond to the coordinates and orientation calculated from the odometry, IMU sensor, and GPS, respectively. Which sensor can produce a minimal E_n, can be determined empirically or by comparing the reading with the other sensors.
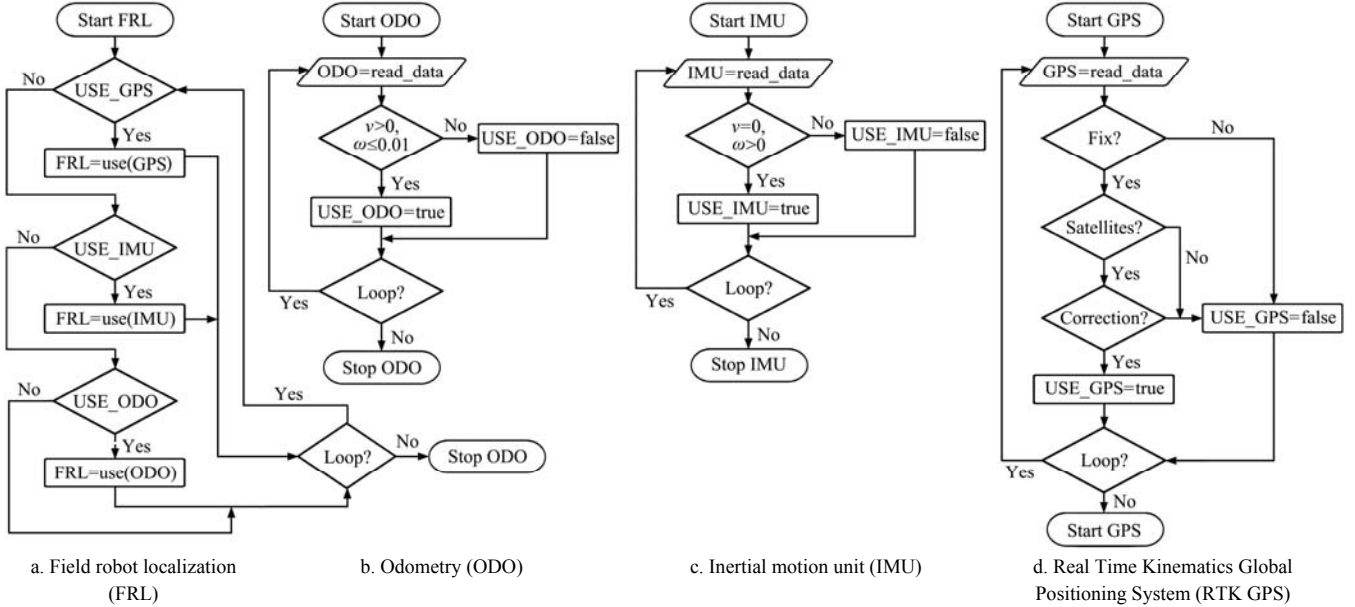

a. Field robot localization (FRL)　　b. Odometry (ODO)　　c. Inertial motion unit (IMU)　　d. Real Time Kinematics Global Positioning System (RTK GPS)

Figure 2　Sensor-specific state machine for choosing the right robot localization sensory system

2.3　Field robot localization algorithm

Well known and widely used localization algorithm in the ROS community was developed by Charles River Analytics[7,8]. However, the reasons why it was not used as part of the Rovitis robot should be made clear. The robot itself is built on different sensory systems that might produce accurate readings most of the time, but there are situations when they fail. In these cases, the approach misses completely and should not be used on a robotic system, including low-cost sensory systems.

So, in order to successfully localize the robot, a custom-made field robot localization (FRL) algorithm was developed that uses three different sensory systems in combination with information regarding the linear and angular speeds set by the path or row following algorithms. The information regarding wheel movement, and odometry, is captured by REP200[23] and connected to Phidgets high-speed encoder[24], inertial information is provided by Phidgets spatial IMU[18], and Navsat Pixi RTK-GPS[18] for the satellite navigation part. The RTK-GPS system used default settings and measurements from the GPS and GLONASS satellites. As explained in the introduction, none of these sensors is accurate enough to produce good localization results on their own. This is the reason why a sensor-specific state machine was implemented to solve this problem and as depicted in Figure 2. The following paragraphs summarize the situations where each sensory system is acceptable at a time and which one should be temporarily discarded.

The odometry-based system works well when the robot is moving straight. So, it is used to calculate the position of the robot when linear speed is more than 0 and angular equal to 0 or close to 0 if no other sensor system is available at that time. If the robot is turning, the odometry is disabled due to the nature of the skid steer system, which causes the slipping of the wheels/tracks used for moving and turning the robot.

The low-cost inertial unit on the robot works great when the internal combustion engine of the robot is now working, but when it is, it is greatly affected by the vibrations caused by the engine. The vibrations cause noise, and the readings drift over time, even if the robot is not moving. This means that, in general, the readings from this sensor would be rejected, but as the robot lacks information regarding the orientation when rotating, it can be used differentially in a short period of time to calculate the orientation of the robot for a short time frame when there is no alternative. Once the other sensors produce good enough readings, the estimates of the IMU sensor are fixed, and the sensor is reset.

Navsat Pixi offers an accurate RTK-GPS system that, in most situations, works well. The system, of course, has to have a fix and has to receive accurate information; when the receiver on the robot is connected to the base station, with no occlusions due to buildings or any other obstacles, when enough satellites are present, and when there is no outside interference, e.g., Signal to Noise Ratio (SNR) is low. This can be monitored by looking at the statuses produced by the Pixi system and is used to calculate the position, in all cases when available, and orientation when the robot is moving on a pseudo-straight path. The system is, however, disabled when the robot is not moving as the GPS locations randomly move around a correct position, which produces wrong calculations of the orientation.

3　Results and discussion

In order to evaluate the field localization system, the robot was

driven manually by remote control to teach it and then repeated in the drive in autonomous mode. The location was chosen with the intention to give the sensors the worst possible conditions for accurate localization, so, the drive took place behind a big metal-enclosed building, positioned on the far left from the robot starting point, and on a sand-covered surface. The height of the building party occluded the GPS base station, and its metallic parts interfered with the magnetic readings of the IMU, while the sand-covered surface caused additional errors in the odometry.

Figure 3 shows the movement of the robot at three different positions with a clear path in the sand that was made when the robot was taught what to do. The first image depicts the starting position with an orientation of 1.0°, the second in movement with an orientation of 95.5°, and the third after completing a rough half of the path with a current orientation of 187.0° compared to starting orientation. These orientations and current positions were calculated via images taken by a hovering DJI MAVIC 2 drone (DJI, PRC) to determine the real robot's position and orientation.

| a. At 1.0° | b. At 95.5° | c. At 187.0° |

Figure 3   Images taken by a DJI MAVIC 2 drone   used as ground truth with three different orientations of the robot

Three different sensory systems produce independent position and/or orientation estimates, where the goal is to get the best possible precision with a real-time localization system needed for subsequent steps like path following.

The first step to evaluate the approach was to show a problem with the low-cost sensors and use their readings with the usually applied robot localization from the ROS repository[7,8]. In order to ensure the same conditions for robot localization and field robot localization, a BAG file[25] was recorded while the robot was driving in autonomous mode, and they were replayed to capture the data from the two algorithms. Figure 4 depicts the results of the robot localization algorithm presented by the software RVIZ.
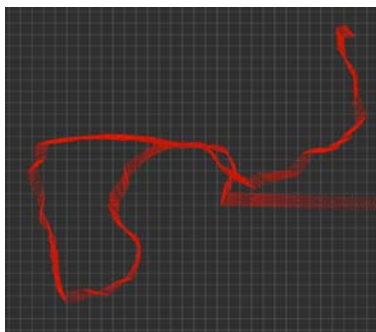
Note: The start of the red arrow represents the position and the orientation of the red arrow the orientation of the robot.

Figure 4   Results of the robot localization as presented by RVIZ

The robot localization from Figure 4 starts in the middle of the map and continues to go the right, but once it gets too close to the building (located at the lower right corner of the pictures from Figure 3) behind the GPS correction signal coverage, it starts to show effects caused by the bad GPS signal, metallic object interferences, vibration caused drifts of the IUM and slipping on the wheels on the sand during the left turn. This completely turns the orientation and positions of the robot and even positions the robot going the wrong way.

In the next step, the field robot localization is evaluated by using the same bag file and providing the same conditions as for the robot localization. In order to assess the real position of the mobile robot in the outdoor test environment, a ground truth reading is provided from the video recording made by a DJI MAVIC 2 drone with a 4K camera, which was hovering above the

robot while it was performing the test as shown in Figure 3.

The ground truth data regarding the actual position and orientation of the robot were calculated from each video frame with the help of OpenCV's template matching algorithm[26], where 720 templates were prepared and used to compare it with each frame to determine the best possible match and the right position/orientation of the moving robot. This provided a per-pixel accuracy that corresponds to 0.012 m in metric dimensions and 0.5° accuracy of orientation.

The comparison of the field robot localization with the ground truth data is shown in Figures 5 and 6 for orientation and position respectively. The blue line presents the readings from the field robot localization, while the green line represents the ground truth data. By comparing the data, it can be concluded that the algorithm made an error of 0.005 m±0.220 m in estimating the position and 0.6°±3.5° when estimating orientation. The average position and orientation data were low, while the standard orientational deviation was rather high, which could be explained by an initial offset of the drone not facing exactly 0° like the robot.

The orientation estimate was initially produced by the last saved good orientation and then adjusted according to the new GPS fixes as long GPS was accurate enough (iterations around 220), then it switched to differential IMU readings (iterations around 320) and back to GPS. Around iterations 480-600 the situation repeated as the robot was driving close to the trees that occluded a clear view of the base station.
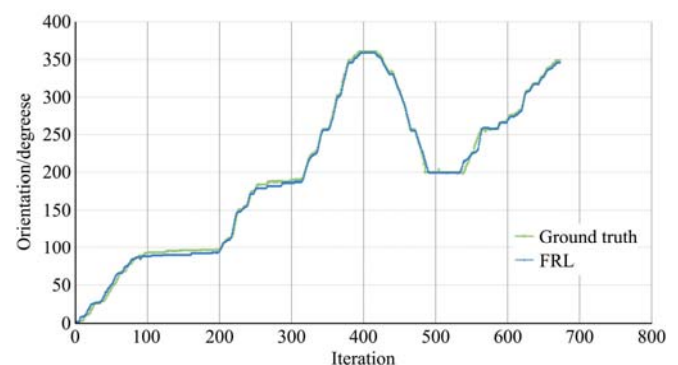

Figure 5   Orientation of the robot calculated with the FRL versus the ground truth data
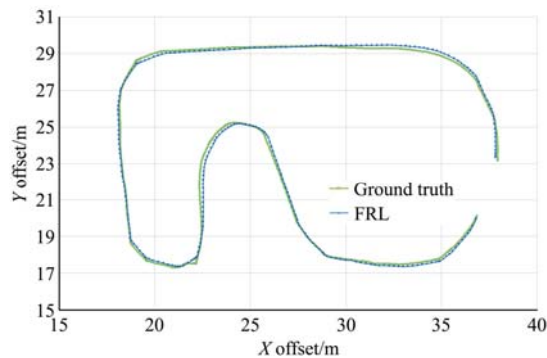
Figure 6　Position of the robot calculated with the FRL versus the ground truth data

The position of the robot was initially produced by the last saved good position and then adjusted according to the new GPS fixes as long GPS was accurate enough (second left turn, upper left part of the figure), then it switched to odometry while the robot was driving straight (lower left side of the figure) and combined IMU to complete the turn. When the robot completed the third left turn it got accurate GPS fixes (small step at the end of the third turn). Similar to the orientation from Figure 5, the situation around the final straight movement, is followed by the last turn.

5　Conclusions

The usual approach when building a robust robotic solution is to use a high-grade, high-cost sensory system. In this study, a different approach was investigated with low-cost sensors to make the solution more commercially accessible and available to a wider range of users that can afford it. This may lead to problems if the sensors fail in some situations, which results in an inaccurate localization. To solve this, the presented approach used data from all the available sensors, including wheel odometry, inertial-motion data from the IMU, and a location fix from an RTK GPS, where the challenge of localizing the robot is solved with a sensor fusion algorithm that works by selecting the most accurate readings in a given moment to produce a precise pose estimation.

The sensor fusion approach was based on a straightforward state machine that chose which of the readings for which of the sensors should be used at a given moment. This can produce a robust enough mechanism but with low-cost sensors that can, as shown in Section 3, outperform some of the widely used approaches in robot localization.

Currently, the developed approach is being tested on two vineyard robots beyond the evaluation test presented in this study in day-to-day operation. One of the possibilities of improvement that could be seen is to include the readings from the LiDAR sensor and improve the accuracy with the help of Simultaneous Localization and Mapping (SLAM) algorithms which will be investigated in the future.

[References]

[1] Lely Astronaut. Robotic milking system. Available: https://pdf.agriexpo.online/pdf/lely/lely-astronaut/169577-7538.html. Accessed on [2020-10-28].

[2] Pilz K H, Feichter S. How robots will revolutionize agriculture. Journal of Science and Technology, 2017; 4: 3437.

[3] Drone mapping and analytics for agriculture. Available: https://www.precisionhawk.com/agriculture. Accessed on [2020-10-28].

[4] Fennimore S A, Cutulle M. Robotic weeders can improve weed control options for specialty crops. Pest Management Science, 2019; 75(7): 1767–1774.

[5] Durmuş H, Güneş E O, Kırcı M, Üstündağ B B. The design of general purpose autonomous agricultural mobile-robot: "AGROBOT". In: 2015 Fourth International Conference on Agro-Geoinformatics (Agro-geoinformatics), Istanbul: IEEE, 2015; pp.49-53. doi: 10.1109/Agro-Geoinformatics.2015.7248088.

[6] Orchard and Vineyard Coverage. Available: https://asirobots.com/farming/orchard-vineyard/. Accessed on [2020-10-30].

[7] Moore T, Stouch D. A generalized extended Kalman filter implementation for the robot operating system. Advances in Intelligent Systems and Computing, 2015; 302: 335–348.

[8] ROS-Robot Localization. Available: http://docs.ros.org/en/noetic/api/robot_localization/html/index.html. Accessedo on [2020-11-1].

[9] Vursavus K K, Yurtlu Y B, Diezma-Iglesias B, Lleo-Garcia L, Ruiz-Altisent M. Classification of the firmness of peaches by sensor fusion. Int J Agric & Biol Eng, 2015; 8(6): 104–115.

[10] Huang S D, Dissanayake G. Robot localization: An introduction. Wiley Encyclopedia of Electrical and Electronics Engineering, 2016; W8318. doi: 10.1002/047134608X.W8318.

[11] Elmenreich W. An introduction to sensor fusion. Technische Universität Wien, Institut für Technische Informatik, 2002; 28p.

[12] Yang Q H, Chang C, Bao G J, Fan J, Xun Y. Recognition and localization system of the robot for harvesting Hangzhou White Chrysanthemums. Int J Agric & Biol Eng, 2018; 11(1): 88–95.

[13] Shalal N, Low T, McCarthy C, Hancock N. Orchard mapping and mobile robot localisation using on-board camera and laser scanner data fusion - Part B: Mapping and localisation, Computers and Electronics in Agriculture, 2015; 119: 267–278.

[14] Chen X Y, Wang S A, Zhang B Q, Luo L. Multi-feature fusion tree trunk detection and orchard mobile robot localization using camera/ultrasonic sensors. Computers and Electronics in Agriculture, 2018; 147: 91–108.

[15] Nemec D, Šimak V, Janota A, Hruboš M, Bubenikova E. Precise localization of the mobile wheeled robot using sensor fusion of odometry, visual artificial landmarks and inertial sensors. Robotics and Automation Systems, 2019; 112: 168–177.

[16] Rovitis Veneto. Available: https://www.aziendapantano.it/rovitis40.html. Accessed on [2020-10-30].

[17] Bianco P, Bellucci V, De Falco C, Deromedis S, Gentilini P, Carlo J, et al. Note sull'inquinamento da pesticidi in Italia (Notes on Italian pesticide pollution). GRE Lazio, European Consumers, 2017; 120p.

[18] Phidgets Spatila 3/3/3 IMU. Available: https://www.phidgets.com/ ?tier= 3&prodid=48. Accessed on [2020-10-30].

[19] Sick LMS111 LiDAR. Available: https://www.sick.com/ag/en/detection-and-ranging-solutions/2d-lidar-sensors/lms1xx/lms111-10100/p/p109842. Accessed on [2020-10-30].

[20] Velodyne VLP16 LiDAR. Available: https://velodynelidar.com/products/puck/. Accessed on [2020-10-30].

[21] Switnav Piksi RTK-GPS. Available: https://www.swiftnav.com/piksi-multi. Accessed on [2020-10-30].

[22] Quigley M, Conley K, Gerkey B P, Faust J, Foote T, Leibs J, er al. ROS: an open-source robot operating system. ICRA Workshop on Open Source Software, 2009; pp.1–6.

[23] Rep200 encoder. Available: https://www.elap.it/incrementalencoders/encoder-rep/. Accessed on [2020-10-30].

[24] Phidgets high speed encoder. Available: https://www.phidgets.com/?tier =3&catid=4&pcid=2&prodid=51. Accessed on [2020-10-30].

[25] Martinez A, Fernandez E. Learning ROS for Robotics Programming. Packet Publishing Ltd., Bermingham, UK, 2013; 332p.

[26] Kaehler A, Bradski G. Learning OpenCV 3 computer vision in C++ with the OpenCV Library. Sebastopol: O'Reilly, 2016, pp.397–406.